

PATENT ABSTRACTS OF JAPAN

(11)Publication number : 07-006043

(43)Date of publication of application : 10.01.1995

(51)Int.Cl. G06F 9/46
G06F 13/00

(21)Application number : 06-034742 (71)Applicant : MITSUBISHI ELECTRIC
CORP

(22)Date of filing : 04.03.1994 (72)Inventor : MIZUNUMA ICHIRO
SHIMAKAWA HIROMITSU

(30)Priority

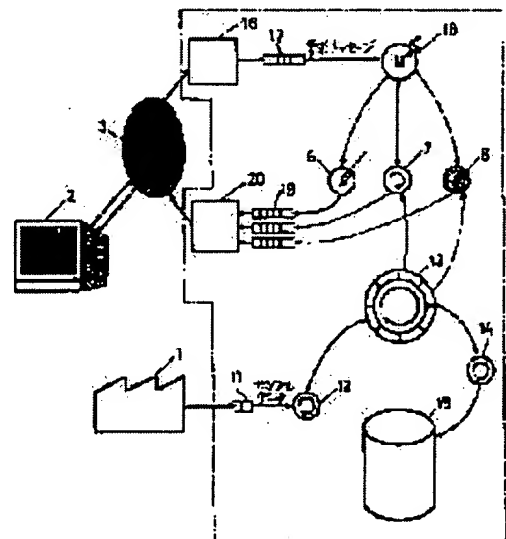
Priority number : 05 69055 Priority date : 05.03.1993 Priority country : JP

(54) MULTITHREAD SERVER

(57)Abstract:

PURPOSE: To perform processing s for request messages even while received sample data are stored in a memory, and to speed up the processing s and prevent a request message from being absent by taking out the request messages, held by a 2nd reception part, in order and performing the processing s corresponding to their contents on the basis of the sample data which are received by a 1st reception part and stored in the memory.

CONSTITUTION: The request messages are sent at irregular intervals from a client 2 through



a network 3 and received by a receiver 16. Then the request messages received by the receiver 16 are temporarily stored in an FIFO queue 17. A mother thread 18 takes in the stored request messages in order and generates and actuates one of child threads 6, 7, and 8 according to the contents. The actuated child thread 6, 7, or 8 performs the processing corresponding to the contents of the request messages on the basis of the sample data stored in a ring buffer 13 and on a disk 15.

LEGAL STATUS

[Date of request for examination] 04.03.1994

[Date of sending the examiner's
decision of rejection] 24.06.1997

[Kind of final disposal of application
other than the examiner's decision of
rejection or application converted
registration]

[Date of final disposal for application]

[Patent number]

[Date of registration]

[Number of appeal against examiner's
decision of rejection]

[Date of requesting appeal against
examiner's decision of rejection]

[Date of extinction of right]

Copyright (C); 1998,2003 Japan Patent Office

(51)Int.Cl. ⁸	識別記号	庁内整理番号	F I	技術表示箇所
G 0 6 F 9/46	3 4 0 B	8120-5B		
13/00	3 5 7 Z	7368-5B		

審査請求 有 請求項の数10 O L (全 24 頁)

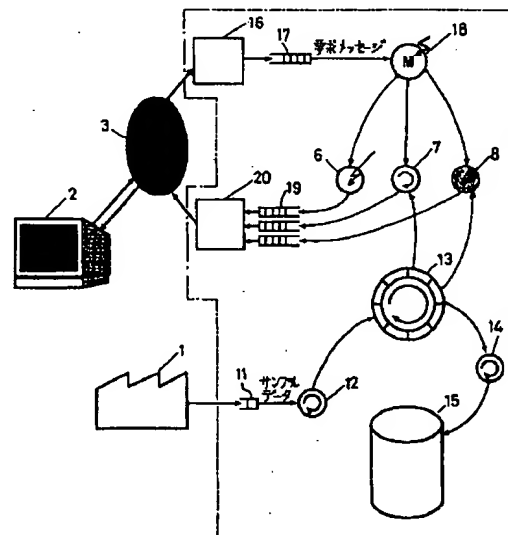
(21)出願番号	特願平6-34742	(71)出願人	000006013 三菱電機株式会社 東京都千代田区丸の内二丁目2番3号
(22)出願日	平成6年(1994)3月4日	(72)発明者	水沼 一郎 尼崎市塚口本町8丁目1番1号 三菱電機 株式会社産業システム研究所内
(31)優先権主張番号	特願平5-69055	(72)発明者	島川 博光 尼崎市塚口本町8丁目1番1号 三菱電機 株式会社産業システム研究所内
(32)優先日	平5(1993)3月5日	(74)代理人	弁理士 田澤 博昭 (外1名)
(33)優先権主張国	日本 (J P)		

(54)【発明の名称】 マルチスレッド・サーバ

(57)【要約】

【目的】 受信したサンプルデータをメモリに格納中であつても、要求メッセージに対する処理を可能にして、要求メッセージに対する処理の高速化を図ることができるとともに、要求メッセージの欠損を防止できるマルチスレッド・サーバを得ることを目的とする。

【構成】 第2の受信部16・17により保持された要求メッセージを順次取り出し、その要求メッセージの内容に応じた処理を第1の受信部11が受信してメモリ13・15に格納したサンプルデータに基づいて行うようにしたものである。



- 1: フラント
2: クライアント (外部装置)
3: ネットワーク (外部装置)
6,7,8: チャイルドスレッド (処理実行部)
11: FIFO キュー (第1の受信部)
13: リングバッファ (メモリ)
15: データク (メモリ)
16: レシーバ (第2の受信部)
17: FIFO キュー (第2の受信部)
18: データスレッド (処理実行部)
19: FIFO キュー (送信部)
20: トランスミッタ (送信部)

【特許請求の範囲】

【請求項1】 プラント等からサンプルデータを受信する第1の受信部と、上記第1の受信部により受信されたサンプルデータを格納するメモリと、外部装置から要求メッセージを受信するとともに、その要求メッセージを保持する第2の受信部と、上記第2の受信部により保持された要求メッセージを順次取り出し、その要求メッセージの内容に応じた処理を上記メモリに格納されたサンプルデータに基づいて行う処理実行部と、上記処理実行部の処理結果を一時的に保持したのち、その処理結果を上記外部装置に対して送信する送信部とを備えたマルチスレッド・サーバ。

【請求項2】 プラント等からサンプルデータを受信する第1の受信部と、上記第1の受信部により受信されたサンプルデータを格納するメモリと、外部装置から要求メッセージを受信するとともに、その要求メッセージを保持する第2の受信部と、上記第2の受信部により保持された要求メッセージを順次取り出し、その要求メッセージの内容に応じた処理を上記メモリに格納されたサンプルデータに基づいて行う処理実行部と、上記処理実行部の処理結果を一時的に保持するとともに、複数の処理結果を保持した場合には優先度の高い処理結果から順次上記外部装置に対して送信する送信部とを備えたマルチスレッド・サーバ。

【請求項3】 上記送信部は、上記処理実行部の処理結果を上記外部装置に対して送信する処理を行う際に、他の処理と競合する場合、該他の処理より優先度が高いときのみ処理を行うことを特徴とする請求項1または請求項2記載のマルチスレッド・サーバ。

【請求項4】 上記処理実行部は、上記第2の受信部が複数の要求メッセージを保持している場合、優先度の高い要求メッセージから順次取り出すことを特徴とする請求項1または請求項2記載のマルチスレッド・サーバ。

【請求項5】 上記処理実行部は、各スレッドが処理を実行する際に必要となるメモリ領域を各スレッド毎に予め確保しておき、各スレッドに処理を実行させる際に対応するメモリ領域を割り当てることを特徴とする請求項1または請求項2記載のマルチスレッド・サーバ。

【請求項6】 上記処理実行部は、処理に必要なスレッドを予め生成し、待機状態にしておくことを特徴とする請求項1または請求項2記載のマルチスレッド・サーバ。

【請求項7】 上記メモリは、上記第1の受信部により受信されたサンプルデータを格納中に、上記処理実行部から所定のアドレスに格納されているサンプルデータの読み込み要求があった場合、その読み込み要求に係るアドレスが、現在サンプルデータを格納しているアドレスと一致するときのみその読み込み要求を却下することを特徴とする請求項1または請求項2記載のマルチスレッド・サーバ。

【請求項8】 上記処理実行部が生成するスレッドのち、周期的に起動される周期起動型スレッドの起動及び終了を管理するタイマスレッドを設け、そのタイマスレッドは所定の起動周期毎にその周期起動型スレッドを起動させるとともに、その周期起動型スレッドが起動後所定時刻経過しても処理を終了しない場合には、その処理を中断させて獲得している資源を解放させることを特徴とする請求項1または請求項2記載のマルチスレッド・サーバ。

【請求項9】 上記処理実行部が生成するスレッドと上記タイマスレッド間の共有資源に対する読み書きの相互排除を行うためにセマフォを設けるとともに、上記タイマスレッドより優先度の低いスレッドがそのセマフォを獲得しているためにそのタイマスレッドが待ち状態になっている場合、一時的にその優先度の低いスレッドの優先度をそのタイマスレッドの優先度と同一にする優先度継承機能を上記セマフォに持たせたことを特徴とする請求項8記載のマルチスレッド・サーバ。

【請求項10】 上記処理実行部が生成するスレッドと上記タイマスレッド間の共有資源に対する読み書きの相互排除を行うために、当該共有資源を使用しているか否かを示すフラグ及びそのフラグを読み書きするためのセマフォを各スレッド毎に設けるとともに、上記タイマスレッドより優先度の低いスレッドが当該セマフォを獲得しているためにそのタイマスレッドが待ち状態になっている場合、一時的にその優先度の低いスレッドの優先度をそのタイマスレッドの優先度と同一にする優先度継承機能を上記セマフォに持たせたことを特徴とする請求項8記載のマルチスレッド・サーバ。

【発明の詳細な説明】

【0001】

【産業上の利用分野】 この発明は、外部装置からの要求メッセージに対し、リアルタイムに応答できるマルチスレッド・サーバに関するものである。

【0002】

【従来の技術】 図22は従来のマルチスレッド・サーバを示す構成図であり、図において、1はプラント、2はクライアント、3はネットワーク（バックプレーンバスでもよい）であり、クライアント2及びネットワーク3から外部装置が構成されている。4はプラント1からサンプルデータを受信するとともに、外部装置から要求メッセージを受信するレシーバ、5はレシーバ4により受信された要求メッセージの内容に応じてチャイルドスレッド6、7、8のいずれかを起動するとともに、レシーバ4がサンプルデータを受信した場合にはそのサンプルデータをメモリ9に格納するマザースレッド、6、7、8は外部装置から送られる要求メッセージに応じた処理をメモリ9に格納されたサンプルデータに基づいて行うチャイルドスレッドであり、チャイルドスレッド6、7、8は起動時にマザースレッド5によって生成され

る。9はサンプルデータを格納するメモリ、10はチャイルドスレッド6, 7, 8の処理結果を一時的に保持するFIFOキュー、20はFIFOキュー10に保持された処理結果を、保持された順に取り出して外部装置へ送信するトランスミッタである。

【0003】次に動作について説明する。まず、レシーバ4は、プラント1からサンプルデータまたは外部装置から要求メッセージを受信する。そして、マザースレッド5は、レシーバ4がプラント1からサンプルデータを受信した場合には、そのサンプルデータをメモリ9に格納し、レシーバ4が外部装置から要求メッセージを受信した場合には、その要求メッセージの内容に応じてチャイルドスレッド6, 7, 8のいずれかを生成して起動する。

【0004】例えば、マザースレッド5によりチャイルドスレッド6が起動された場合には、チャイルドスレッド6は、要求メッセージに応じた処理をメモリ9に格納されたサンプルデータに基づいて行う。そして、チャイルドスレッド6の処理結果がFIFOキュー10に送信され、一時的に保持される。そして、トランスミッタ20はFIFOキュー10に保持された処理結果を、保持された順に取り出して外部装置へ送信する。即ち、図23に示すように、FIFOキュー10に先に保持された処理結果が、後に保持された処理結果より優先度が低い場合でも、優先度の低い先に保持された処理結果から送信される。なお、チャイルドスレッド6, 7, 8はそれぞれ並列的に処理を行うことができるが、メモリ9に対するアクセスは同時に行うことができない。

【0005】

【発明が解決しようとする課題】従来のマルチスレッド・サーバは以上のように構成されているので、プラント1等からのサンプルデータも外部装置からの要求メッセージもレシーバ4が受信してマザースレッド5がその後の処理を決定しなければならず、従って、例えば、受信した要求メッセージに回答する処理を行っている途中で、サンプルデータを受信した場合、その処理が終了するまでそのサンプルデータをメモリ9に格納することができず、その結果、そのサンプルデータの格納が次のサンプルデータの到着までに終了しないと、サンプルデータの欠測が起こる事態が発生する問題点があった。

【0006】また、レシーバ4は、複数の要求信号等を保持する機能がないので、マザースレッド5が、レシーバ4により受信された要求メッセージ等（便宜上、旧要求メッセージという）を取り込む前に、他の要求メッセージ等を外部装置から送られてしまうと、旧要求メッセージが欠損してしまう問題点があった。

【0007】また、トランスミッタ20は、FIFOキュー10に保持された順に取り出して送信するので、先に保持された処理結果より後から保持された処理結果の方が高速性を要求されていても、先に保持された処理結

果を送信するまで後に保持された処理結果は待たされてしまい、外部装置の要求を十分満足することができない問題点があった。

【0008】また、トランスミッタ20は、送信以外の他の処理を行うスレッド（図3参照）の存在を確認せずに処理結果を送信してしまうので、トランスミッタ20の送信処理より当該スレッドの処理の方が高速性を要求されていても、トランスミッタ20の送信処理が終わるまで当該スレッドの処理が待たされてしまう問題点があった。

【0009】また、マザースレッド5は、要求メッセージに回答する処理を実行する際に、チャイルドスレッド6, 7, 8を生成する毎にチャイルドスレッド6, 7, 8の処理に必要なメモリ領域を確保しなければならず、従って、徐々にメモリ空間上に不要となったメモリ領域が増えていくためガベージコレクションを行う必要が生じ、その間要求メッセージに対する回答ができなくなる問題点があった。

【0010】また、マザースレッド5は、要求メッセージに回答する処理を実行する際に、その都度チャイルドスレッド6, 7, 8を生成する必要があるが、スタック領域が不足しているような場合には生成することができず、スタック領域が解放されるまで処理を待機しなければならない問題点があった。

【0011】さらに、チャイルドスレッド6, 7, 8はそれぞれ並列的に処理を行うことができるが、メモリ9に対するアクセスは同時に行うことができないので、並列処理の効率が低下し、高速処理ができないなどの問題点もあった。

【0012】請求項1の発明は上記のような問題点を解消するためになされたもので、受信したサンプルデータをメモリに格納中であっても、要求メッセージに対する処理を可能にして、要求メッセージに対する処理の高速化を図ることができるとともに、要求メッセージの欠損を防止できるマルチスレッド・サーバを得ることを目的とする。

【0013】請求項2の発明は、上記目的に加え、高速性の要求される処理結果から順次送信できるようにして、外部装置の要求を十分満足できるマルチスレッド・サーバを得ることを目的とする。

【0014】請求項3の発明は、処理結果を外部装置へ送信する処理よりも、高速性の要求される処理がある場合には、その高速性の要求される処理を優先して処理ができるマルチスレッド・サーバを得ることを目的とする。

【0015】請求項4の発明は、要求メッセージの欠損を防止できるとともに、高速性の要求される要求メッセージに対する処理から行うことができるマルチスレッド・サーバを得ることを目的とする。

【0016】請求項5の発明は、ガベージコレクション

を行う必要をなくし、要求メッセージに対する応答を妨げられることのないマルチスレッド・サーバを得ることを目的とする。

【0017】請求項6の発明は、スタック領域の不足を理由に、要求メッセージに対する処理が遅延するのを防止できるマルチスレッド・サーバを得ることを目的とする。

【0018】請求項7の発明は、並列処理の効率を向上させ、高速に処理を行うことができるマルチスレッド・サーバを得ることを目的とする。

【0019】請求項8の発明は、周期的に処理を行う周期起動型スレッドのリアルタイム性を確保できるマルチスレッド・サーバを得ることを目的とする。

【0020】請求項9及び請求項10の発明は、タイマスレッドの待ち時間をできるだけ短くできるマルチスレッド・サーバを得ることを目的とする。

【0021】

【課題を解決するための手段】請求項1の発明に係るマルチスレッド・サーバは、第2の受信部により保持された要求メッセージを順次取り出し、その要求メッセージの内容に応じた処理を第1の受信部が受信してメモリに格納したサンプルデータに基づいて行うようにしたものである。

【0022】請求項2の発明に係るマルチスレッド・サーバは、処理実行部の処理結果を一時的に保持するとともに、複数の処理結果を保持した場合には優先度の高い処理結果から順次外部装置に対して送信するようにしたものである。

【0023】請求項3の発明に係るマルチスレッド・サーバは、処理実行部の処理結果を外部装置に対して送信する処理を行う際に、他の処理と競合する場合、該他の処理より優先度が高いときのみ処理を行うようにしたものである。

【0024】請求項4の発明に係るマルチスレッド・サーバは、第2の受信部が複数の要求メッセージを保持している場合、優先度の高い要求メッセージから順次取り出すようにしたものである。

【0025】請求項5の発明に係るマルチスレッド・サーバは、各スレッドが処理を実行する際に必要となるメモリ領域を各スレッド毎に予め確保しておき、各スレッドに処理を実行させる際に対応するメモリ領域を割り当てるようにしたものである。

【0026】請求項6の発明に係るマルチスレッド・サーバは、処理に必要なスレッドを予め生成し、待機状態にしておくようにしたものである。

【0027】請求項7の発明に係るマルチスレッド・サーバは、第1の受信部により受信されたサンプルデータをメモリに格納中に、処理実行部から所定のアドレスに格納されているサンプルデータの読み込み要求があった場合、その読み込み要求に係るアドレスが、現在サン

プルデータを格納しているアドレスと一致するときのみその読み込み要求を却下するようにしたものである。

【0028】請求項8の発明に係るマルチスレッド・サーバは、所定の起動周期毎に周期起動型スレッドを起動させるとともに、その周期起動型スレッドが起動後所定時刻経過しても処理を終了しない場合には、その処理を中断させて獲得している資源を解放させるタイマスレッドを設けたものである。

【0029】請求項9の発明に係るマルチスレッド・サーバは、タイマスレッドより優先度の低いスレッドがセマフォを獲得しているためにそのタイマスレッドが待ち状態になっている場合、一時的にその優先度の低いスレッドの優先度をそのタイマスレッドの優先度と同一にする優先度継承機能をそのセマフォに持たせたものである。

【0030】請求項10の発明に係るマルチスレッド・サーバは、共有資源を使用しているか否かを示すフラグ及びそのフラグを読み書きするためのセマフォを各スレッド毎に設けるとともに、タイマスレッドより優先度の低いスレッドが当該セマフォを獲得しているためにそのタイマスレッドが待ち状態になっている場合、一時的にその優先度の低いスレッドの優先度をそのタイマスレッドの優先度と同一にする優先度継承機能をそのセマフォに持たせたものである。

【0031】

【作用】請求項1の発明におけるマルチスレッド・サーバは、要求メッセージを保持する第2の受信部を設けたことにより、要求メッセージの欠損が防止され、また、第2の受信部により保持された要求メッセージを順次取り出し、その要求メッセージの内容に応じた処理を第1の受信部が受信してメモリに格納したサンプルデータに基づいて行う処理実行部を設けたことにより、第2の受信部から要求メッセージを受信中でも、第1の受信部からサンプルデータを受信してメモリに格納することができる。

【0032】請求項2の発明におけるマルチスレッド・サーバは、処理実行部の処理結果を一時的に保持するとともに、複数の処理結果を保持した場合には優先度の高い処理結果から順次外部装置に対して送信する送信部を設けたことにより、高速性の要求される処理結果から順次送信できるようになる。

【0033】請求項3の発明におけるマルチスレッド・サーバは、処理実行部の処理結果を外部装置に対して送信する処理を行う際に、他の処理と競合する場合、該他の処理より優先度が高いときのみ処理を行う送信部を設けたことにより、処理結果を外部装置へ送信する処理よりも、高速性の要求される処理がある場合には、その高速性の要求される処理から優先して処理される。

【0034】請求項4の発明におけるマルチスレッド・サーバは、第2の受信部が複数の要求メッセージを保持

している場合、優先度の高い要求メッセージから順次取り出す処理実行部を設けたことにより、要求メッセージの欠損を防止され、また、高速性の要求される要求メッセージに対する処理から行なえる。

【0035】請求項5の発明におけるマルチスレッド・サーバは、各スレッドが処理を実行する際に必要となるメモリ領域を各スレッド毎に予め確保しておき、各スレッドに処理を実行させる際に対応するメモリ領域を割り当てる処理実行部を設けたことにより、ガベージコレクションを行う必要がなくなる。

【0036】請求項6の発明におけるマルチスレッド・サーバは、処理に必要となるスレッドを予め生成し、待機状態にしておく処理実行部を設けたことにより、スタック領域の不足を理由に、要求メッセージに対する処理が遅延するのを防止される。

【0037】請求項7の発明におけるマルチスレッド・サーバは、第1の受信部により受信されたサンプルデータをメモリに格納中に、処理実行部から所定のアドレスに格納されているサンプルデータの読み込み要求があった場合、その読み込み要求に係るアドレスが、現在サンプルデータを格納しているアドレスと一致するときのみその読み込み要求を却下するようにしたことにより、同じデータを同時に読み書きすることによって生じるデータ内容の矛盾を防止しつつデータの読み書きの並列処理が可能になる。

【0038】請求項8の発明におけるマルチスレッド・サーバは、所定の起動周期毎に周期起動型スレッドを起動させるとともに、その周期起動型スレッドが起動後所定時刻経過しても処理を終了しない場合には、その処理を中断させて獲得している資源を解放させるタイマスレッドを設けたことにより、他の周期起動型スレッドを所定の起動周期毎に確実に起動できるようになる。

【0039】請求項9の発明におけるマルチスレッド・サーバは、タイマスレッドより優先度の低いスレッドがセマフォを獲得しているためにそのタイマスレッドが待ち状態になっている場合、一時的にその優先度の低いスレッドの優先度をそのタイマスレッドの優先度と同一にする優先度継承機能をそのセマフォに持たせたことにより、優先度の低いスレッドがセマフォを獲得していても、その優先度の低いスレッドの処理が速やかに行われるため、タイマスレッドがセマフォを獲得するまでの待ち時間が短縮される。

【0040】請求項10の発明におけるマルチスレッド・サーバは、共有資源を使用しているか否かを示すフラグ及びそのフラグを読み書きするためのセマフォを各スレッド毎に設けるとともに、タイマスレッドより優先度の低いスレッドが当該セマフォを獲得しているためにそのタイマスレッドが待ち状態になっている場合、一時的にその優先度の低いスレッドの優先度をそのタイマスレッドの優先度と同一にする優先度継承機能をそのセマフ

ォに持たせたことにより、優先度の低いスレッドがセマフォを獲得していても、その優先度の低いスレッドの処理が速やかに行われるため、タイマスレッドがセマフォを獲得するまでの待ち時間が短縮される。

【0041】

【実施例】

実施例1. 以下、この発明の一実施例を図について説明する。図1はこの発明の実施例1によるマルチスレッド・サーバを示す構成図であり、図において、従来のものと同一符号は同一または相当部分を示すので説明を省略する。11はプラント1からサンプルデータを受信するFIFOキュー（第1の受信部）、12はFIFOキュー11が受信したサンプルデータを一定周期でリングバッファ13に書き込む周期起動型スレッド、13はそのサンプルデータを格納するリングバッファ（メモリ）、14は周期起動型スレッド12が現在書き込み中であるリングバッファ13のインデックス（アドレス）より少し過去のサンプルデータをいくつかまとめてディスク15へ一定周期で書き込む周期起動型スレッド、15はディスク（メモリ）である。

【0042】また、16はクライアント2からネットワーク3を介して要求メッセージを受信するレシーバ、17はレシーバ16が受信したその要求メッセージを保持するFIFOキューであり、レシーバ16及びFIFOキュー17から第2の受信部が構成されている。

【0043】また、18はFIFOキュー17により保持された要求メッセージを順次取り出し、その要求メッセージの内容に応じてチャイルドスレッド6、7、8のいずれかを起動することにより、その要求メッセージの内容に応じた処理をリングバッファ13及びディスク15に格納されたサンプルデータに基づいて行うマザースレッドであり、マザースレッド18及びチャイルドスレッド6、7、8から処理実行部が構成されている。

【0044】また、19はチャイルドスレッド6、7、8の処理結果を一時的に保持するFIFOキュー、20はFIFOキュー19により保持されたその処理結果をネットワーク3を介してクライアント2に対して送信するトランスミッタであり、FIFOキュー19及びトランスミッタ20から送信部が構成されている。

【0045】次に動作について説明する。まず、サンプルデータは、例えば、10msec毎にプラント1から送信され、FIFOキュー11がこれを受信する。そして、FIFOキュー11に受信されたサンプルデータは、周期起動型スレッド12によって周期的にリングバッファ13に格納される。また、リングバッファ13にある程度サンプルデータが蓄積されていくと、周期起動型スレッド14によってサンプルデータをいくつかまとめてディスク15に格納される。なお、リングバッファ13及びディスク15にサンプルデータを格納する際、いつサンプルされたサンプルデータであるか分かるよう

に時刻情報もいっしょに格納される。

【0046】一方、要求メッセージは、不定期にクライアント2からネットワーク3を介して送信され、レシーバ16がこれを受信する。そして、レシーバ16に受信された要求メッセージは、一時的にFIFOキュー17に格納される。そして、FIFOキュー17に格納された要求メッセージを、マザースレッド18が順次取り込み、その要求メッセージの内容に応じてチャイルドスレッド6, 7, 8のいずれかを生成して起動する（チャイルドスレッド6, 7, 8を起動する際、処理に必要なメモリ領域を動的に確保する。詳細は後述する。）。ここで、要求メッセージには、下記に示すようなものがある。

- ① あるイベントが起こった場合にそのイベントの発生を通知
- ② 現在時刻のサンプルデータの周期転送
- ③ 過去のある時刻からある時刻までのサンプルデータの一斉転送

【0047】そして、マザースレッド18により起動されたチャイルドスレッド6, 7, 8は、要求メッセージの内容に応じた処理をリングバッファ13及びディスク15に格納されているサンプルデータに基づいて行う

（必要とするサンプルデータがリングバッファ13に格納されていない場合、ディスク15とアクセスする）。例えば、要求メッセージの内容が、上記①の内容であれば、イベントの起動があった旨の通知をクライアント2に対して送信する処理を行う。また、要求メッセージの内容が、上記②③の内容であれば、リングバッファ13等から該当するサンプルデータを読み込んでクライアント2に対して送信する処理を行う。

【0048】そして、チャイルドスレッド6, 7, 8の処理結果はFIFOキュー19に送信され、一時的に保持される。なお、チャイルドスレッド6, 7, 8は並列処理が可能であるため、FIFOキュー19に複数の処理結果が保持されることがある。そして、FIFOキュー19に保持された処理結果は、トランスミッタ20が順次取り込んでネットワーク3を介してクライアント2に対して送信し、一連の処理を終了する。

【0049】このように、実施例1では、要求メッセージを受信するレシーバ16の他に、プラント1から送信されるサンプルデータを受信する専用のFIFOキュー11を設けるようにしているので、要求メッセージの処理と別個独立にサンプルデータのサンプル処理を行うことができるようになり、従って、要求メッセージの処理によってサンプル処理が妨げられる事態が発生することがなくなり、サンプル処理をその時間上の制約を保って行うことができる。また、要求メッセージを一時的に保持するFIFOキュー17を設けているので、前に送られた要求メッセージに対する処理が終了する前に他の要求メッセージが複数個クライアント2から送信されてき

たとしても、要求メッセージが欠損するという事態は発生しない。因に、この実施例1は請求項1の発明に対応している。

【0050】実施例2. 上記実施例1では、FIFOキュー19に複数の処理結果を保持された場合、FIFOキュー19に保持された順にその処理結果を送信するものについて説明したが、図2に示すように、複数の処理結果を保持した場合には優先度の高い処理結果から順次クライアント2に対して送信するようにしてもよい。図2において、19a、19b、19cはFIFOキュー19の構成要素であり、それぞれチャイルドスレッド6, 7, 8の処理結果を一時的に保持する。また、21はFIFOキュー19が複数の処理結果を保持した場合には優先度の高い処理結果から順次クライアント2に対して送信するトランスミッタ（送信部）である。

【0051】次に動作について説明する。まず、要求メッセージは、上記のごとく内容がそれぞれ異なり、従って、それぞれクライアント2に対して応答するに際して時間上の制約に関係した優先度が存在する。例えば、チャイルドスレッド6は、上記①の要求メッセージに対する処理を行うものである場合、かかる処理は、イベント発生に対するクライアント2の対応が間に合うように、ある時間以内にクライアント2に送信される必要があるので極めて高い優先度 P_{mq2} を有している。また、チャイルドスレッド7は、上記②の要求メッセージに対する処理を行うものである場合、かかる処理は、クライアント2がまだ最新のサンプルデータとして利用する価値のある時間以内に送信される必要があるので、チャイルドスレッド6の処理に次いで高い優先度 P_{mq1} を有する。また、チャイルドスレッド8は、上記③の要求メッセージに対する処理を行うものである場合、かかる処理は、送信が多少遅れても支障が少ないのでチャイルドスレッド6, 7の処理より低い優先度 P_{mq0} を有している。

$$P_{mq0} < P_{mq1} < P_{mq2}$$

【0052】この場合において、チャイルドスレッド8から優先度 P_{mq0} の処理結果がFIFOキュー19cに送信されて格納された後、直ちにチャイルドスレッド6から優先度 P_{mq2} の処理結果がFIFOキュー19aに送信されて格納されたとする。実施例1では、FIFOキュー19に格納された順に送信するので、チャイルドスレッド8の処理結果を先に送信することになるが、実施例2では、処理結果の優先度を考慮して送信するので、優先度の高いチャイルドスレッド6の処理結果を先に送信することになる。これにより、クライアント2の要求に則した処理が可能になる。因に、この実施例2は請求項2の発明に対応している。

【0053】実施例3. 上記実施例2では、トランスミッタ21は、送信以外の他の処理を行うスレッド22の存在を確認せずに処理結果を送信するものについて説明

したが、図3に示すように、チャイルドスレッド6, 7, 8の処理結果をクライアント2に対して送信する処理を行う際に、送信以外の他の処理を行うスレッド22と競合する場合、スレッド22の処理の優先度 P_{etc} より高い優先度をトランスミッタ21の送信スレッドが有するときのみ送信処理を行うようにしてもよい。図において、21a, 21b, 21cはトランスミッタ21の送信スレッド、22は送信以外の他の処理を行うスレッド、23はセマフォである。

【0054】次に動作について説明する。例えば、送信スレッド21a, 21b, 21cの優先度をそれぞれ P_{tr2} , P_{tr1} , P_{tr0} とし、優先度の高さの関係は下記のようにあるとする。

$$P_{tr0} < P_{tr1} < P_{etc} < P_{tr2}$$

【0055】この場合において、送信スレッド21aが送信処理を行うときは、送信スレッド21aの優先度 P_{tr2} は、スレッド22の優先度 P_{etc} より高いので、スレッド22が存在していても直ちに送信処理が行われる。一方、送信スレッド21cが送信処理を行うときは、送信スレッド21cの優先度 P_{tr0} は、スレッド22の優先度 P_{etc} より低いので、スレッド22の処理が終了するまで送信処理が待機される。

【0056】なお、セマフォ23は、複数の送信スレッド21a, 21b, 21cが同時にネットワーク3へ送信するのを避けるために設けられている。セマフォ23を獲得できる送信スレッドは1つであり、セマフォ23を獲得している送信スレッド(図3の例では送信スレッド21aが獲得している)のみがネットワーク3に対して送信することができる。

【0057】ここで、セマフォ23を設けた場合下記の不具合がある。即ち、低い優先度を有する送信スレッド21cがセマフォ23を獲得した後に、高い優先度を有する送信スレッド21aがFIFOキュー19aに格納されたような場合、低い優先度を有する送信スレッド21cがセマフォ23を獲得しているので、高い優先度を有する送信スレッド21aは送信スレッド21cの送信処理を待たなければならないが、送信スレッド21cの優先度はスレッド22の優先度より低いので、送信スレッド21cはスレッド22の処理が終了するまで待機状態になる。従って、スレッド21aは高い優先度を有しているにもかかわらず、結果として、スレッド22及び送信スレッド21cの処理が終了するまで送信処理を行うことができない不具合がある。

【0058】そこで、セマフォ23に優先度継承機能(送信スレッドの優先度変更機能)を持たせることにより上記不具合を解消することができる。即ち、一時的に送信スレッド21cの優先度を送信スレッド21aの優先度まで高めることにより、送信スレッド21cの送信処理をスレッド22に優先させる。送信スレッド21cの処理が終了すれば、送信スレッド21aがセマフォ2

3を獲得し、送信スレッド21aの優先度はスレッド22の優先度より高いので、送信スレッド21aはスレッド22の処理を待たずに送信が可能になる。従って、送信スレッド21aの待ち時間を短縮することができる。因に、この実施例3は請求項3の発明に対応している。

【0059】実施例4. 上記実施例1では、要求メッセージがFIFOキュー17に保持された順に、マザースレッド18が取り出して処理するものについて説明したが、図4に示すように、要求メッセージに優先度を持たせ、FIFOキュー17に先に格納された要求メッセージの優先度より、後に格納された要求メッセージの優先度が高い場合、後に格納された要求メッセージからマザースレッド18が取り込んで処理するようにしてもよい。これにより、高速性の要求される要求メッセージに対する処理から行うことができるようになる。因に、この実施例4は請求項4の発明に対応している。

【0060】実施例5. 上記実施例1では、要求メッセージの内容に応じてマザースレッド18がチャイルドスレッド6, 7, 8のいずれかを生成して起動する際、処理に必要なメモリ領域を動的に確保するものについて説明したが、図5に示すように、各チャイルドスレッド6, 7, 8が処理を実行する際に必要となるメモリ領域を各スレッド6, 7, 8毎に予め確保しておき、各スレッド6, 7, 8に処理を実行させる際に対応するメモリ領域を割り当てるようにしてもよい。また、上記実施例1では、マザースレッド18がその都度チャイルドスレッド6, 7, 8を生成するようにしていたが、チャイルドスレッド6, 7, 8を予め生成して待機状態にしておいてもよい。

【0061】以下、図5について詳細に説明する。図5において、31はスレッドプール、32, 33, 34はそれぞれチャイルドスレッド6, 7, 8が処理を行う際に必要となるメモリ領域であり、スレッドを実行すべき関数へのポインタ、その関数への引数、スレッドの優先度、周期起動型スレッドの場合には、周期及びデッドライン、イベント起動型スレッドの場合には、待つべきイベント(以下、起動イベントという)の種類などの情報が格納される。なお、各チャイルドスレッド6, 7, 8は初期化時には待機状態にある。

【0062】また、35はスレッドプール31を管理するスレッドプール管理テーブル、36は管理テーブルセマフォ、37は待機中のチャイルドスレッド6, 7, 8の数を格納する領域、38はマザースレッド18の起動要求をスレッドプール31に対して知らせる要求セマフォ、39は起動要求を受けつけたチャイルドスレッド6, 7, 8のスレッドIDを格納する領域、40はスレッドプール31からマザースレッド18へ要求の受領を知らせるための受領セマフォ、41はマザースレッド18からスレッドプール31へ起動命令を伝えるための起動セマフォである。また、42はマザースレッド18以

外にスレッドプール31を用いるスレッドである。

【0063】次に動作について説明する。まず、マザースレッド18は、スレッドプール管理テーブル35へのアクセス権を得るために管理テーブルセマフォ36にロックをかける（動作1）。そして、管理テーブルセマフォ36へのロックに失敗した場合には、一定時間後に再度ロックを試みる。管理テーブルセマフォ36へのロックに成功した場合には、領域37に格納されている待機中のスレッドを確認し（動作2）、“0”であれば起動不可能として例外処理を行うか、あるいは、一定時間後に再度起動を試みる。“1”以上であれば要求セマフォ38に対してシグナルを送信する（動作3）。

【0064】また、待機中のチャイルドスレッド6、8はすべて要求セマフォ38からのシグナルの到着を待っており、マザースレッド18からのシグナルは、これらの待機中のチャイルドスレッド6、8のいずれか1つによって受信される。この例の場合では、チャイルドスレッド6によって受信されたものとする。

【0065】チャイルドスレッド6は、起動を受領したスレッドが自分であることをマザースレッド18に知らせるため、自分のスレッドIDを領域39に書き込み（動作5）、その後、受領セマフォ40にシグナルを送信し（動作6）、起動セマフォ41へのシグナルの到着を待つ。

【0066】一方、マザースレッド18は、動作3の終了後、受領セマフォ40へのシグナルの到着を待っており、チャイルドスレッド6からのシグナルを受信すると（動作7）、領域39から要求を受領したスレッドIDを読み取る（動作8）。

【0067】ここで、マザースレッド18は、予め確保されているメモリ領域32に対して起動に必要な情報を書き込み（動作9）、書き込みが完了すると、起動セマフォ41へシグナルを送信する（動作10）。チャイルドスレッド6は、起動セマフォ41からシグナルを受信すると（動作11）、領域37の待機中のスレッドの数を“1”減らし（動作12）、管理テーブルセマフォ36へのロックを解除する（動作13）。

【0068】このように、予め各チャイルドスレッド6、7、8が実行時に必要となるメモリ領域32、33、34を確保しておくことにより、メモリ領域確保の繰り返しを原因とするメモリ空間上の不要なメモリが増加していくという事態は発生しなくなる。従って、ガベージコレクションを行う必要もなくなる。また、予め各チャイルドスレッド6、7、8を初期化時に生成しておくことにより、スタック領域の不足を原因とするチャイルドスレッド6、7、8の生成不可という事態は発生しなくなる。従って、スタック領域が解放されるまで処理を待機しなければならないという従来の問題点は解消される。因に、この実施例5は請求項5及び請求項6の発明に対応している。

【0069】次に、リアルタイム性を要求されないチャイルドスレッドのスレッドプール31内の構成及び動作について説明する。図6はスレッドプール31内のチャイルドスレッドが有する起動情報の構成図である。実行すべき関数へのポインタ、その関数への起動要求をだしたマザースレッド18からの引数、実行の優先度が含まれている。また、図7はスレッドプール31内のチャイルドスレッドの動作を示すフローチャートであり、マザースレッド18、あるいは他のスレッド42から終了の割り込みがかかると、ステップST11に制御を移す。

【0070】次に、周期起動型スレッドのスレッドプール31内の構成及び動作について説明する。図8はスレッドプール31内のチャイルドスレッドが有する起動情報の構成図である。一周期ごとに実行すべき関数へのポインタ、その関数への起動要求をだしたマザースレッド18からの引数、実行の優先度、起動の周期、一周期あたりの処理のデッドラインが含まれている。また、図9はスレッドプール31内のチャイルドスレッドの動作を示すフローチャートであり、マザースレッド18、あるいは他のスレッド42が実行中の周期起動型スレッドを停止させる場合には、各周期起動型スレッドの持つ終了フラグを立てる。各周期起動型スレッドはその一周期ごとに終了フラグをチェックし、それが立っていれば処理を終了し、待機状態に戻る。

【0071】次に、イベント起動型スレッドのスレッドプール31内の構成及び動作について説明する。図10はスレッドプール31内のチャイルドスレッドが有する起動情報の構成図である。イベントが発生したときに実行すべき関数へのポインタ、その関数への起動要求をだしたマザースレッド18からの引数、実行の優先度、起動イベントに関する情報、処理のデッドラインが含まれている。また、図11はスレッドプール31内のチャイルドスレッドの動作を示すフローチャートである。マザースレッド18、あるいは他のスレッド42がイベント待ちをしているイベント起動型スレッドを待機状態にする場合には、そのスレッドに対して終了イベントを送信する。終了イベントを受信したスレッドは待機状態に戻る。

【0072】スレッドプール31内の各チャイルドスレッド6、7、8に、リアルタイム性を要求されないスレッド、周期起動型スレッド、イベント起動型スレッドのいずれの動作をも行わせるようにすることができる。マザースレッド18からスレッドプール31内の各チャイルドスレッド6、7、8へ受け渡す起動情報の中に起動するスレッドの種類を含ませる。スレッドプール31内のチャイルドスレッド6、7、8はこの情報に基づき行う処理を決定する。また、図12はスレッドプール31内の各チャイルドスレッド6、7、8がもつ起動情報の構成図であり、図13はスレッドプール31内の各チャイルドスレッドの動作を示すフローチャートである。

【0073】ここで、スレッドプール31内の数を n とする。クライアント2から送られるスレッドの起動要求（要求メッセージ）によってこれらのスレッドに処理が受け渡される。あるイベントが生じた場合の緊急通知を行うスレッドの数の上限を n_1 、現在起動されている数を i_1 、サンプルデータの周期的な転送を行うスレッドの数の上限を n_2 、現在起動されている数を i_2 、あるイベントが生じた場合の任意期間のサンプルデータの転送を行うスレッドの数の上限を n_3 、現在起動されている数を i_3 とする。また、下記の関係があるとする。

$$n_1 + n_2 + n_3 > n$$

$$n_1 \leq n$$

$$n_2 \leq n$$

$$n_3 \leq n$$

【0074】この場合において、 i_1 、 i_2 が小さく、下記の関係がある場合には、任意期間のサンプルデータの転送を行うスレッドを最大数 n_3 まで起動することができる。

$$i_1 + i_2 + i_3 \leq n \quad \dots (1)$$

そして、緊急通知を行うスレッドや周期転送を行うスレッドの起動要求がクライアント2から送られて式(1)の関係を満たさなくなった場合には、これを満たすように任意期間のサンプルデータの転送を行うスレッドのいくつかを中断あるいは終了させる。

【0075】実施例6. 図14はリングバッファ13及びディスク15に対して複数のスレッドが書き込み及び読み出しを行う動作を説明する構成図である。以下、リングバッファ13等をアクセスするスレッドの機能に応じて説明する。

【0076】① 書き込みスレッド

まず、周期起動型スレッド12（書き込みスレッド）は、リングバッファ13へ周期ごとに書き込みを行う。書き込まれるリングバッファ13のインデックスは書き込みごとに一つずつ移動する。ここで、周期起動型スレッド12が現在書き込みを行っているインデックス（図中、×で示してあるインデックス）を格納する領域51が予め用意されている。また、領域51は複数のスレッドからアクセスされるため、領域51の相互排除のためセマフォ52が設けられている。

【0077】そして、周期起動型スレッド12は書き込み中に、他のスレッドが書き込み中のインデックスにアクセスするのを防止すべく、書き込むインデックスを変えるごとに、領域51にそのインデックスを書き込む。なお、各スレッドの領域51へのアクセスに要する時間は小さく、またセマフォ52に優先度継承機能を持たせているので、各スレッドがセマフォ52の獲得のために長い時間待たされることはない。

【0078】② バックアップスレッド

周期起動型スレッド14（バックアップスレッド）はリングバッファ13上のサンプルデータを、それらが失わ

れないうちにディスク15へ格納する。ディスク15上にはバックアップ用のファイルがいくつか用意され、各スレッドからは図に示すように、リング状に配置されているように見ることができる。周期起動型スレッド14は、リングバッファ13上に対して周期起動型スレッド12が書き込んでいるインデックスのサンプルデータよりある程度遅れた時刻のサンプルデータを一度にいくつかまとめてディスクに転送する。

【0079】周期スレッド12及び周期スレッド14がアクセスするリングバッファ13上のインデックスの移動の速度は等しく、また、周期スレッド14は周期スレッド12が書き込みを行っているインデックスから十分はなれたインデックスを読み出すので、周期スレッド12及び周期スレッド14が同じインデックスを同時にアクセスすることはない。また、ディスク15上のファイルには、各ファイルに定められた容量まで書き込み、その容量に達するとそのファイルをクローズし、次のファイルへの書き込みを始める。

【0080】ここで、ディスク15に対しても、現在書き込みを行っているファイルに関する領域53を用意するとともに、領域53の相互排除のためセマフォ54を設けている。領域53の格納する情報は、現在書き込みが行われているファイルの番号とそのファイル上にある最も過去のサンプルデータの時刻である。従って、周期起動型スレッド14は、現在書き込みを行っているファイルの番号を領域53に書き込むようにしている。また、新しいファイルに書き込みを始める際には、そのファイルに書き込んだ最初のサンプルデータの時刻を領域53に書き込むようにしている。

【0081】③ 周期的な読み出しを行うスレッド

例えば、チャイルドスレッド7が周期起動型スレッドの場合、チャイルドスレッド7は、一周周期ごとに、リングバッファ13上に格納されている最新のサンプルデータを読み出す。チャイルドスレッド7はその周期ごとに領域51の内容を読み出し、現在周期起動型スレッド12が書き込みを行っているインデックスの1つ手前のインデックス（図中、○で示してあるインデックス）の内容を読み出す。

【0082】④ 任意期間の読み出しを行うスレッド

例えば、チャイルドスレッド8が任意期間の読み出しを行うスレッドの場合、チャイルドスレッド8は、任意時刻から任意時刻までのサンプルデータを任意の間隔で読み出す。要求されたサンプルデータのうちリングバッファ13及びディスク15上にないものがある場合には、リングバッファ13及びディスク15上にあるサンプルデータのみを読み出しに先行して、そのサンプルデータをリングバッファ13から読み出すべきか、ディスク15から読み出すべきかを決定する。

【0083】領域53に格納されている現在書き込まれているファイル上の最も過去のサンプルデータの時刻

と、読み出すべきサンプルデータの時刻を比較する。後者の方が過去のものであれば、既にディスク15上から失われているほどの過去のサンプルデータでない限り、ディスク15から読み出す。後者が前者と同じ、あるいは後者の方が前者より新しい時刻であれば、領域51の示すインデックスに格納されているサンプルデータより新しい時刻でない限り、リングバッファ13から読み出す(図中、△で示してあるインデックス)。

【0084】このように、実施例6によれば、FIFOキュー11により受信されたサンプルデータをリングバッファ13に格納中に、チャイルドスレッド7、8から所定のアドレスに格納されているサンプルデータの読み込み要求があった場合、現在サンプルデータを格納しているアドレスよりも手間のアドレスの内容に限り読み出せるようにしているので、即ち、読み込み要求に係るアドレスが現在サンプルデータを格納しているアドレスと一致するときはその読み込み要求を却下することによって生じるデータ内容の矛盾を防止しつつデータの読み書きの並列処理が可能になり、その結果、高速に処理を行うことができる効果がある。因に、この実施例6は請求項7の発明に対応している。

【0085】実施例7。上記実施例1～6では、周期起動型スレッドの処理の終了については特に言及しなかったが、周期起動型スレッドが起動後所定時刻経過しても処理を終了しない場合には、その処理を中断させて獲得している資源を解放させるようにしてもよい。これにより、上記実施例1～6であれば、ある周期起動型スレッドの処理が複雑である等のため処理時間内に処理が終了しない場合には、リアルタイム性を要求される他の周期起動型スレッドの処理の開始が遅れる等の不具合があったが、他の周期起動型スレッドを所定の起動周期毎に確実に起動できるようになる。

【0086】図15はこの発明の実施例7によるマルチスレッド・サーバを示す構成図であり、図において、61はマザースレッド18が生成するチャイルドスレッド6、7、8等のうち、周期的に起動される周期起動型スレッド7、12、14の起動及び終了を管理するタイマスレッドであり、そのタイマスレッド61は所定の起動周期毎にその周期起動型スレッド7、12、14を起動させるとともに、その周期起動型スレッド7、12、14が起動後、所定時刻(デッドライン)経過しても処理を終了しない場合には、その処理を中断させて獲得している資源を解放させる機能を有している。62は各スレッドの情報を格納するスレッド管理テーブル(資源)である(図16参照)。

【0087】次に動作について説明する。まず、スレッド管理テーブル62には、図16に示すように、当該スレッドが周期起動型スレッドであるか否かを示すUフラグ、当該スレッドが周期毎の処理を実行中であるか否か

を示すEフラグ、当該スレッドの起動時刻までの時間をタイマスレッド61の起動回数(例えば、タイマスレッド61の起動周期が100msであって、スレッドの起動時刻までの時間が200msであれば、起動回数は2回となる)で示すUSフィールド(既に起動されている場合には負の値になる)、当該スレッドの次の起動までの時間をタイマスレッド61の起動回数で示すUWフィールド、当該スレッドのデッドラインまでの時間をタイマスレッド61の起動回数で示すUDフィールド、当該スレッドの周期を格納する周期フィールド、当該スレッドのデッドラインを格納するデッドラインフィールドが記憶されている。

【0088】そして、スレッド管理テーブル62に記憶されている各スレッドの情報は、マザースレッド18が管理しており、具体的には、マザータスク18が要求メッセージにしたがってチャイルドスレッドを生成等するときに、そのチャイルドスレッドの情報をスレッド管理テーブル62に登録する。一方、当該スレッドの消滅、停止要求等を受けるとそのスレッドの登録を抹消する。

【0089】従って、スレッド管理テーブル62の記憶内容を解読すれば、どのスレッドが周期起動型スレッドであるか否か、当該スレッドの起動周期、当該スレッドのデッドライン等の情報を得ることができるので、タイマスレッド61は、周期起動型スレッドの起動周期に比べて極めて短い周期毎にスレッド管理テーブル62の記憶内容を解読する。

【0090】そして、タイマスレッド61は、ある周期起動型スレッドが起動時刻に到達していれば、その周期起動型スレッドに対して起動シグナルを通知することにより、その周期起動型スレッドを起動させる。また、その周期起動型スレッドが起動してから所定時刻経過しても処理が終了しない場合には、その周期起動型スレッドにタイムアウトを通知することにより、その周期起動型スレッドの処理を強制的に中断させるとともに、他の周期起動型スレッドの処理を妨げないために、その周期起動型スレッドが獲得している資源を解放させるなどの必要なタイムアウト処理を実行させる。

【0091】このように、実施例7によれば、タイマスレッド61が、周期起動型スレッドが起動後所定時刻経過しても処理を終了しない場合には、その処理を中断させて獲得している資源を解放させるようにしているので、ある周期起動型スレッドの処理が複雑である等のため処理時間内に処理が終了しない場合でも、他の周期起動型スレッドの起動に影響が及ぶことがなく、周期起動型スレッドのリアルタイム性を確保することができる。因に、この実施例7は請求項8の発明に対応している。

【0092】実施例8。上記実施例7では、各スレッド間の共有資源であるスレッド管理テーブル62に対する読み書きの相互排除について特に言及しなかったが、スレッド管理テーブル62は各スレッド間の共有資源であ

るため、複数のスレッドが同一のデータを同時に読み書きする場合が考えられる。そして、同一データの同時読み書きがなされた場合には、その影響でデータ内容に矛盾が生じる場合があるので、読み書きに対して相互排除の必要性がある。例えば、タイマスレッド61がスレッド管理テーブル62の内容を読み込み中に、周期起動型スレッド等がスレッド管理テーブル62の内容を書き換えてしまうと、タイマスレッド61は正しいデータが得られなくなる場合がある。

【0093】そこで、相互排除の方法として、スレッドがセマフォを獲得した場合に限りスレッド管理テーブル62に対して読み書きできるようにする手法が考えられるが、単にセマフォの獲得だけを読み書きできる条件とすると、例えば下記に示すような不具合を生じる場合がある。即ち、タイマスレッド61より優先度の低いスレッド7等がセマフォを獲得しているためにそのタイマスレッド61が待ち状態になっている場合、タイマスレッド61がセマフォを獲得して起動するためには、その優先度の低いスレッド7等が処理を終了してセマフォを解放しなければならぬが、そのスレッド7等は優先度が低いために、他のスレッドの処理が終了するのを待つ待機状態になる場合があり、この場合、スレッド7等は処理を終了することができないためセマフォを解放することができず、その結果、タイマスレッド61が起動できないという不具合を生じることがある。因に、タイマスレッド61は、他のスレッドの起動等を管理するスレッドであるため、マザースレッド18や周期起動型スレッド等よりも優先度が高く設定されている。

【0094】この実施例8は、上記のような不具合を解消しつつスレッド管理テーブル62に対する読み書きの相互排除を行うものであり、実施例8の構成を下記に示す。即ち、この実施例8では、図16に示すように、スレッド管理テーブル62に対する読み書きの相互排除を行うためにセマフォ63、64を設けるとともに、タイマスレッド61より優先度の低いスレッド7等がそのセマフォ63、64を獲得しているためにそのタイマスレッド61が待ち状態になっている場合、一時的にその優先度の低いスレッド7等の優先度をそのタイマスレッド61の優先度と同一にする優先度継承機能をセマフォ63、64に持たせるようにしたものである。

【0095】これにより、本来優先度の低いスレッド7等の優先度が、優先度の高いタイマスレッド61の優先度と一時的に同一になるため、優先度の関係で他のスレッドの処理の実行に伴ってスレッド7等が待機状態になる場合がほとんどなくなり、その結果、スレッド7等の処理が速やかに行われてセマフォ63、64が解放されるので、タイマスレッド61は速やかにセマフォ63、64を獲得できるようになる。従って、周期起動型スレッドのリアルタイム性を確保できるとともに、装置の信頼性が向上する。

【0096】ここで、図17はタイマスレッド61の動作を示すフローチャートであり、その動作を簡単に説明すると、タイマスレッド61は、周期起動型スレッドの起動時刻、周期、デッドラインを表す時間の粒度（最小単位）を周期として起動される。例えば、起動時刻、周期、デッドラインがすべて100ms単位で表されるものであれば、タイマスレッド61は、100ms周期で起動する。そして、タイマスレッド61は各周期毎に起動され、ステップST51でUフラグを読み取るためにセマフォ63を獲得し、ステップST67及びステップST72でEフラグを読み取るためのセマフォ64を獲得する。また、ステップST53でセマフォ63を解放し、ステップST69及びステップST74でセマフォ64を解放する。

【0097】次に、図18はマザースレッド18がスレッド管理テーブル62に周期起動型スレッドを登録する動作を示すフローチャートであり、その動作を簡単に説明すると、ステップST81でセマフォ63を獲得したのちステップST82でUフラグの読み取りを行い、ステップST83でセマフォ63を解放する（以下、ステップST81～83の間をクリティカルセクションU1という）。また、ステップST86でセマフォ63を獲得したのちステップST87でUフラグの書き込み読みを行い、ステップST88でセマフォ63を解放する（以下、ステップST86～88の間をクリティカルセクションU2という）。

【0098】次に、図19はマザースレッド18がスレッド管理テーブル62から周期起動型スレッドの登録を削除する動作を示すフローチャートであり、その動作を簡単に説明すると、ステップST91でセマフォ63を獲得したのちステップST92でUフラグの書き込みを行い、ステップST93でセマフォ63を解放する（以下、ステップST91～93の間をクリティカルセクションU3という）。

【0099】次に、図20は各周期起動型スレッドの動作を示すフローチャートであり、その動作を簡単に説明すると、ステップST103でセマフォ64を獲得したのちステップST104でEフラグの書き込みを行い、ステップST105でセマフォ64を解放する（以下、ステップST103～105の間をクリティカルセクションE1という）。また、ステップST111でセマフォ64を獲得したのちステップST112でEフラグの書き込み読みを行い、ステップST113でセマフォ64を解放する（以下、ステップST111～113の間をクリティカルセクションE2という）。

【0100】以上で明らかのように、タイマスレッド61が起動時に、優先度の低いスレッド（マザースレッド18、周期起動型スレッド）によって待たされる時間は、セマフォ63、64が優先度継承機能を有しているので、最も待たされる場合でも、クリティカルセクショ

ンU1, U2, U3のうちの何れか1つを実行するのに要する時間と、クリティカルセクションE1, E2のうちの何れか一方の実行に要する時間とを加えた時間に限られる。また、その加えた時間は、高々フラグ2個の読み書きに要する時間であって、タイマスレッド61が行うすべての処理に比べて極めて短い時間であり、無視できる程度の時間である。因に、この実施例8は請求項9の発明に対応している。

【0101】実施例9. 上記実施例8では、UフラグまたはEフラグの全体にセマフォ63, 64を設けたものについて示したが、図21に示すように、スレッド管理テーブル62を使用しているか否かを示すSフラグ及びそのSフラグを読み書きするためのセマフォ65を各スレッド毎に設け、そして、タイマスレッド61より優先度の低いスレッド7等が当該セマフォ65を獲得しているためにそのタイマスレッド61が待ち状態になっている場合、一時的にその優先度の低いスレッド7等の優先度をそのタイマスレッド61の優先度と同一にする優先度継承機能をセマフォ65に持たせるようにしてもよく、上記実施例8と同様の効果を奏する。因に、この実施例9は請求項10の発明に対応している。

【0102】

【発明の効果】以上のように、請求項1の発明によれば、第2の受信部により保持された要求メッセージを順次取り出し、その要求メッセージの内容に応じた処理を第1の受信部が受信してメモリに格納したサンプルデータに基づいて行うように構成したので、要求メッセージを受信している途中であっても、サンプルデータを受信してメモリに格納できるようになり、その結果、サンプル処理をその時間上の制約を保って行うことができる効果がある。

【0103】請求項2の発明によれば、処理実行部の処理結果を一時的に保持するとともに、複数の処理結果を保持した場合には優先度の高い処理結果から順次外部装置に対して送信するように構成したので、高速性の要求される処理結果から順次送信できるようになり、その結果、外部装置の要求を十分満足できるようになる効果がある。

【0104】請求項3の発明によれば、処理実行部の処理結果を外部装置に対して送信する処理を行う際に、他の処理と競合する場合、該他の処理より優先度が高いときのみ処理を行うように構成したので、処理結果を外部装置へ送信する処理よりも、高速性の要求される処理がある場合には、その高速性の要求される処理を優先して処理ができるようになる効果がある。

【0105】請求項4の発明によれば、第2の受信部が複数の要求メッセージを保持している場合、優先度の高い要求メッセージから順次取り出すように構成したので、要求メッセージの欠損を防止できるとともに、高速性の要求される要求メッセージに対する処理から行うこ

とができる効果がある。

【0106】請求項5の発明によれば、各スレッドが処理を実行する際に必要となるメモリ領域を各スレッド毎に予め確保しておき、各スレッドに処理を実行させる際に対応するメモリ領域を割り当てるように構成したので、ガベージコレクションを行う必要がなくなり、その結果、ガベージコレクションを原因とする要求メッセージに対する応答が妨げられるという事態が発生しなくなる効果がある。

【0107】請求項6の発明によれば、処理に必要なスレッドを予め生成し、待機状態にしておくように構成したので、スタック領域の不足を理由に、要求メッセージに対する処理が遅延するのを防止できる効果がある。

【0108】請求項7の発明によれば、第1の受信部により受信されたサンプルデータをメモリに格納中に、処理実行部から所定のアドレスに格納されているサンプルデータの読み込み要求があった場合、その読み込み要求に係るアドレスが、現在サンプルデータを格納しているアドレスと一致するときのみその読み込み要求を却下するように構成したので、同じデータを同時に読み書きすることによって生じるデータ内容の矛盾を防止しつつデータの読み書きの並列処理が可能になり、その結果、高速に処理を行うことができる効果がある。

【0109】請求項8の発明によれば、所定の起動周期毎に周期起動型スレッドを起動させるとともに、その周期起動型スレッドが起動後所定時刻経過しても処理を終了しない場合には、その処理を中断させて獲得している資源を解放させるタイマスレッドを設けるように構成したので、他の周期起動型スレッドを所定の起動周期毎に確実に起動できるようになり、周期起動型スレッドのリアルタイム性を確保できる効果がある。

【0110】請求項9の発明によれば、タイマスレッドより優先度の低いスレッドがセマフォを獲得しているためにそのタイマスレッドが待ち状態になっている場合、一時的にその優先度の低いスレッドの優先度をそのタイマスレッドの優先度と同一にする優先度継承機能をそのセマフォに持たせるように構成したので、同じデータを同時に読み書きすることによって生じるデータ内容の矛盾を防止しつつ、タイマスレッドが速やかにセマフォを獲得できる結果、周期起動型スレッドのリアルタイム性を確保できるとともに、装置の信頼性が向上する効果がある。

【0111】請求項10の発明によれば、共有資源を使用しているか否かを示すフラグ及びそのフラグを読み書きするためのセマフォを各スレッド毎に設けるとともに、タイマスレッドより優先度の低いスレッドが当該セマフォを獲得しているためにそのタイマスレッドが待ち状態になっている場合、一時的にその優先度の低いスレッドの優先度をそのタイマスレッドの優先度と同一にす

る優先度継承機能をそのセマフォに持たせるように構成したので、同じデータを同時に読み書きすることによって生じるデータ内容の矛盾を防止しつつ、タイマスレッドが速やかにセマフォを獲得できる結果、周期起動型スレッドのリアルタイム性を確保できるとともに、装置の信頼性が向上する効果がある。

【図面の簡単な説明】

【図 1】この発明の実施例 1 によるマルチスレッド・サーバを示す構成図である。

【図 2】実施例 2 における送信部の詳細を示す構成図である。

【図 3】実施例 3 における送信部の詳細を示す構成図である。

【図 4】実施例 4 における第 2 の受信部の詳細を示す構成図である。

【図 5】スループット機構の構成を示す構成図である。

【図 6】スレッドプール 31 内のチャイルドスレッドが有する起動情報の構成図である。

【図 7】スレッドプール 31 内のチャイルドスレッドの動作を示すフローチャートである。

【図 8】スレッドプール 31 内のチャイルドスレッドが有する起動情報の構成図である。

【図 9】スレッドプール 31 内のチャイルドスレッドの動作を示すフローチャートである。

【図 10】スレッドプール 31 内のチャイルドスレッドが有する起動情報の構成図である。

【図 11】スレッドプール 31 内のチャイルドスレッドの動作を示すフローチャートである。

【図 12】スレッドプール 31 内のチャイルドスレッドが有する起動情報の構成図である。

【図 13】スレッドプール 31 内のチャイルドスレッドの動作を示すフローチャートである。

【図 14】リングバッファ 13 及びディスク 15 に対して複数のスレッドが書き込み及び読み出しを行う動作を説明する構成図である。

【図 15】この発明の実施例 7 によるマルチスレッド・

サーバを示す構成図である。

【図 16】スレッド管理テーブルの内容を説明する説明図である。

【図 17】タイマスレッドの動作を示すフローチャートである。

【図 18】マザースレッドがスレッド管理テーブルに周期起動型スレッドを登録する動作を示すフローチャートである。

【図 19】マザースレッドがスレッド管理テーブルから周期起動型スレッドの登録を削除する動作を示すフローチャートである。

【図 20】各周期起動型スレッドの動作を示すフローチャートである。

【図 21】スレッド管理テーブルの内容を説明する説明図である。

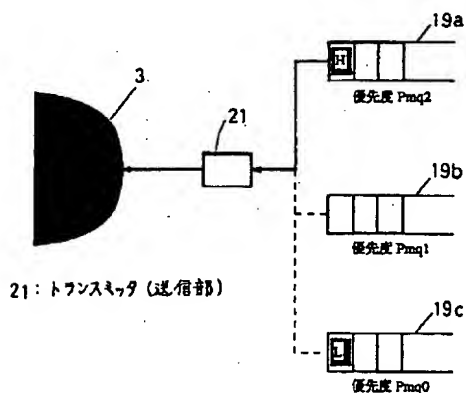
【図 22】従来のマルチスレッド・サーバを示す構成図である。

【図 23】従来の送信部の詳細を示す構成図である。

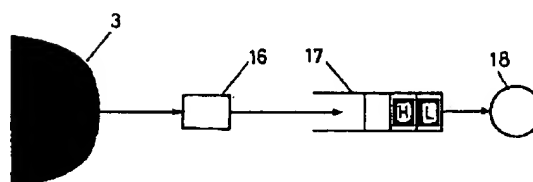
【符号の説明】

- 1 プラント
- 2 クライアント (外部装置)
- 3 ネットワーク (外部装置)
- 6、7、8 チャイルドスレッド (処理実行部)
- 11 FIFOキュー (第 1 の受信部)
- 13 リングバッファ (メモリ)
- 15 ディスク (メモリ)
- 16 レシーバ (第 2 の受信部)
- 17 FIFOキュー (第 2 の受信部)
- 18 マザースレッド (処理実行部)
- 19 FIFOキュー (送信部)
- 20、21 トランスミッタ (送信部)
- 21a、21b、21c 送信スレッド (送信部)
- 32、33、34 メモリ領域
- 61 タイマスレッド
- 62 スレッド管理テーブル (資源)
- 63～65 セマフォ

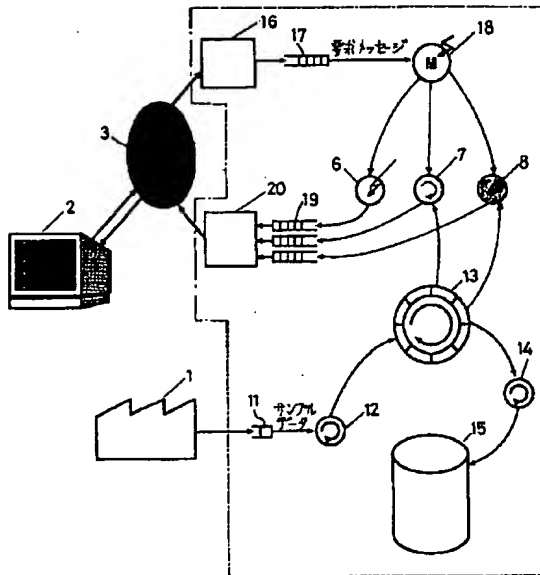
【図 2】



【図 4】

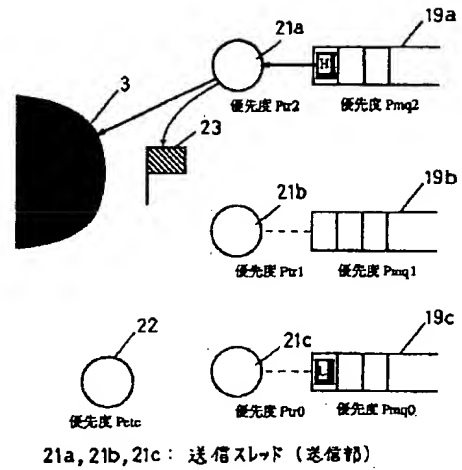


【図 1】

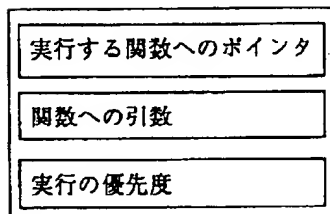


- | | |
|----------------------------|-----------------------|
| 1: アラント | 16: レシーバ (第2の受信部) |
| 2: クライアント (外部装置) | 17: FIFO キュー (第2の受信部) |
| 3: ネットワーク (外部装置) | 18: マサスレッド (処理実行部) |
| 6, 7, 8: チャイルドスレッド (処理実行部) | 19: FIFO キュー (送信部) |
| 11: FIFO キュー (第1の受信部) | 20: トランスマッタ (送信部) |
| 13: リングバッファ (メモリ) | |
| 15: ディスク (メモリ) | |

【図 3】

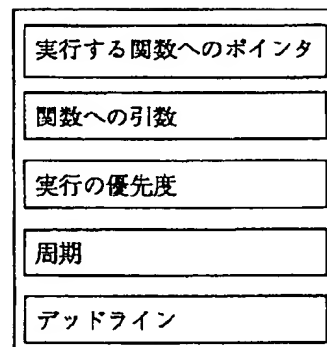


【図 6】



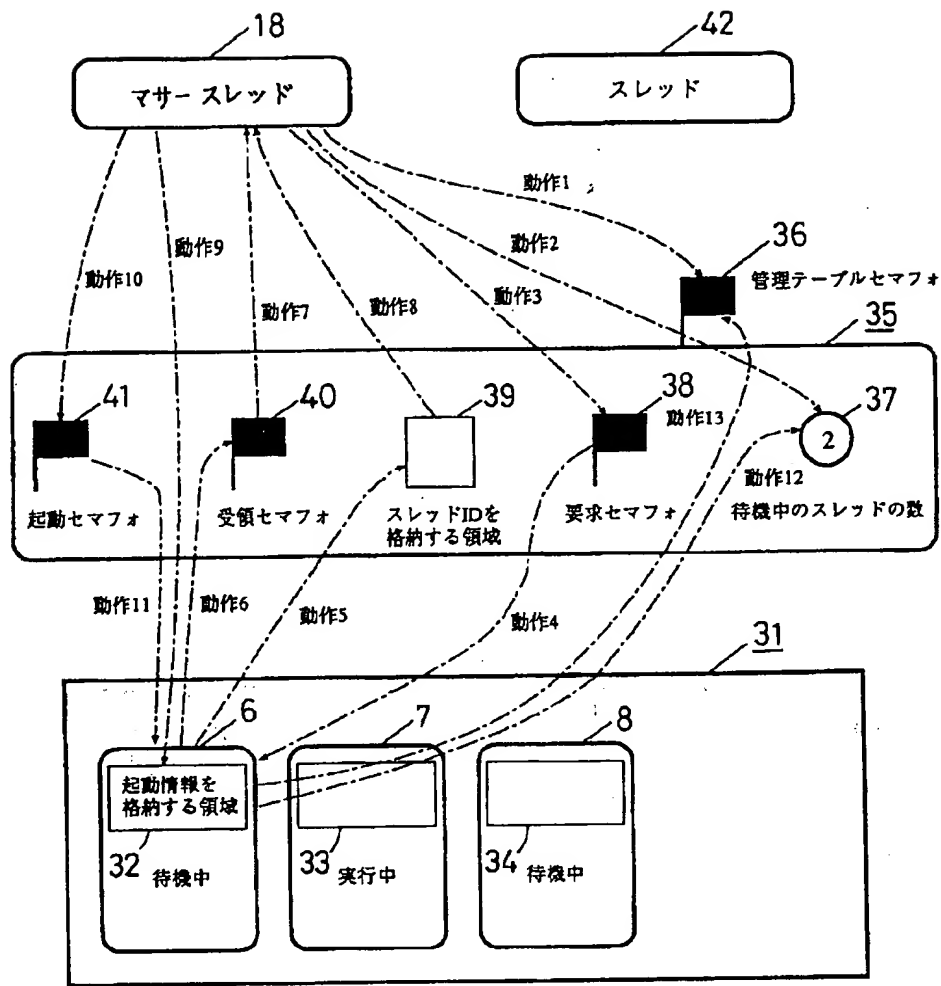
リアルタイム性を要しないスレッドの起動情報

【図 8】



周期起動型スレッドの起動情報

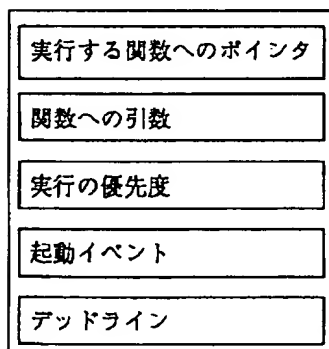
【図5】



32,33,34: メモリ領域

【図10】

【図16】



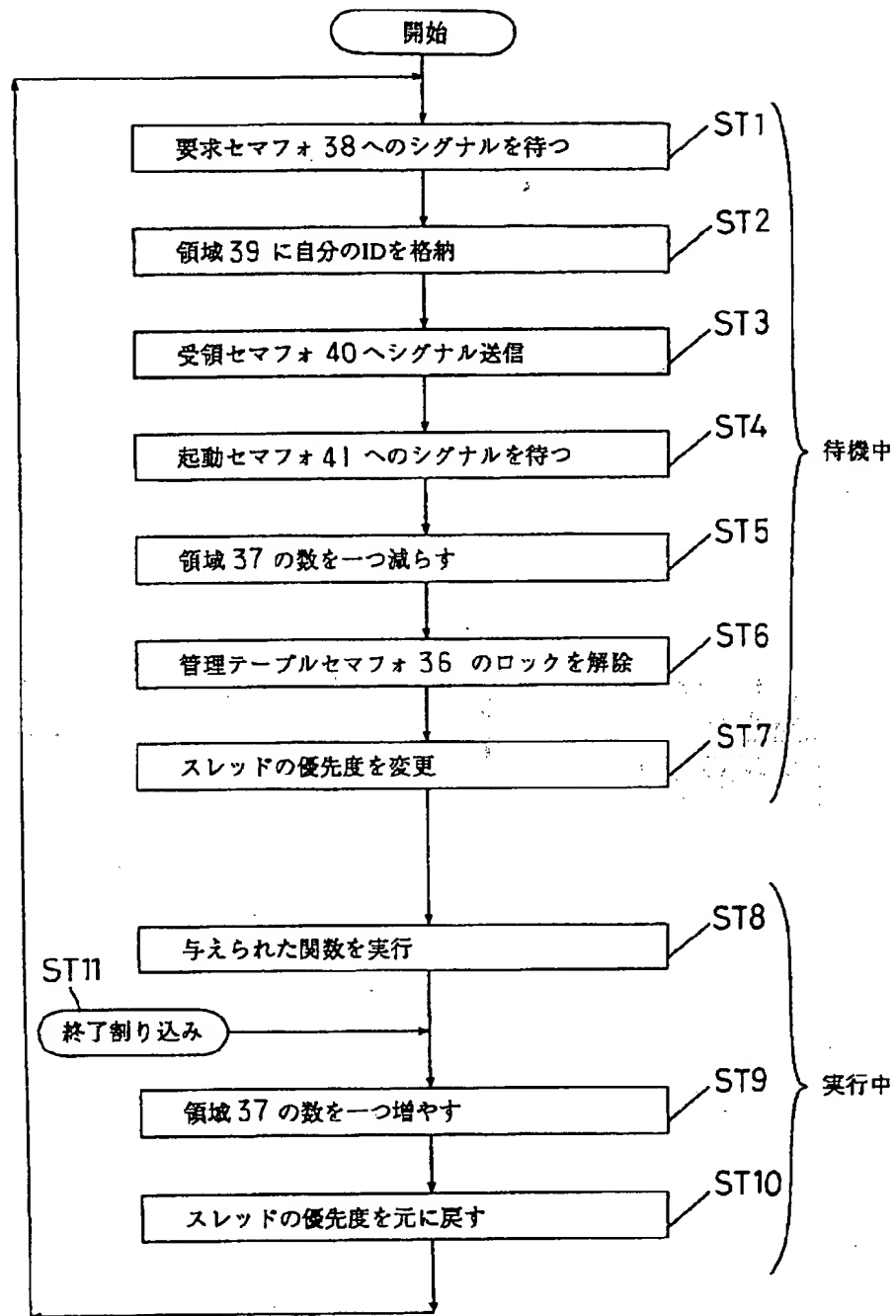
イベント起動型スレッドの起動情報

インデックス	Uフラグ	Eフラグ	US	UW	UD	周知	デッドライン
0	F	—					
1	T	T					
2	T	F					
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
53	T	T					
54	T	F					
55	F	—					
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮

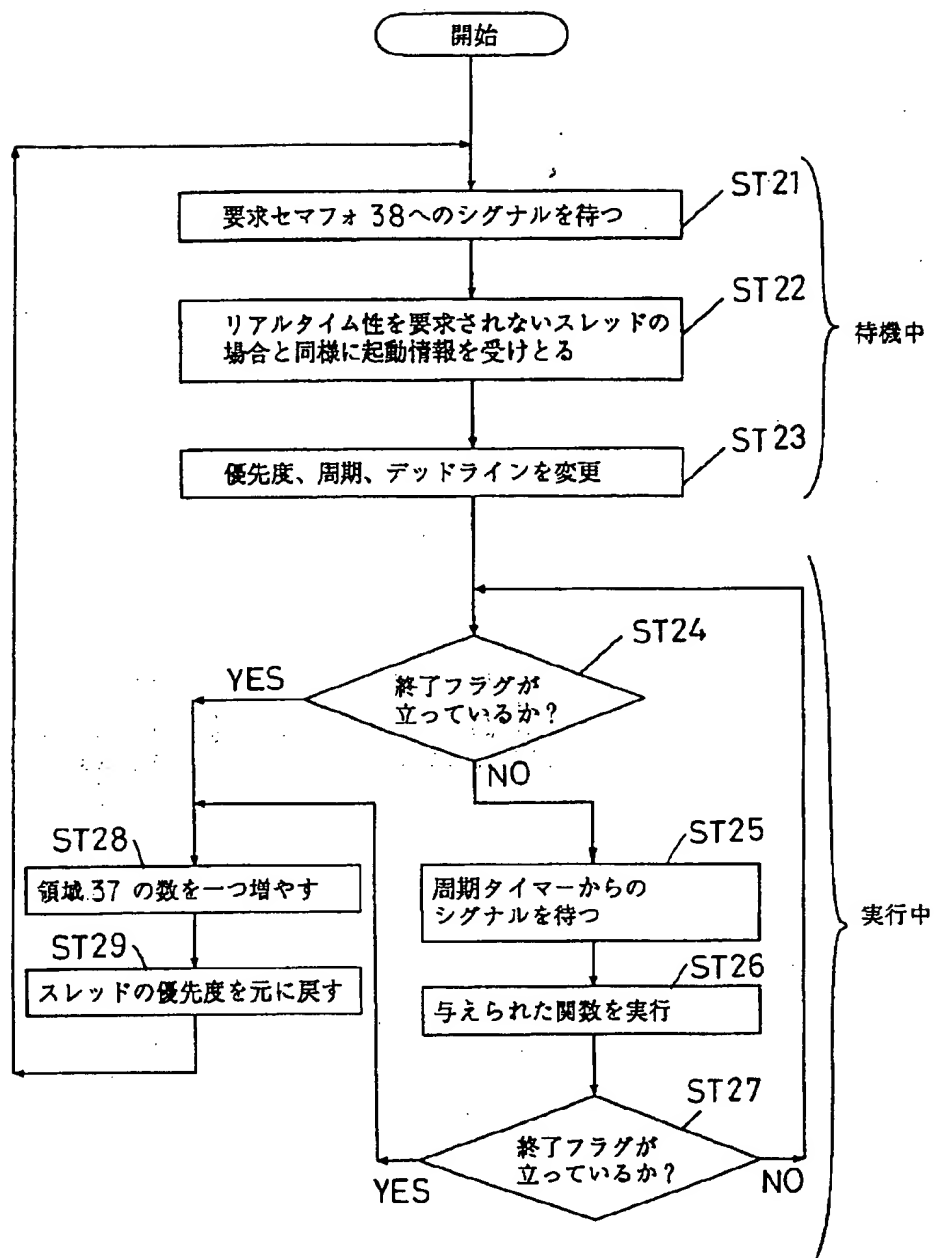
T: TRUE
F: FALSE
—: 任意

63, 64: セマフォ

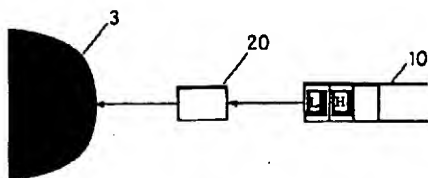
【図7】



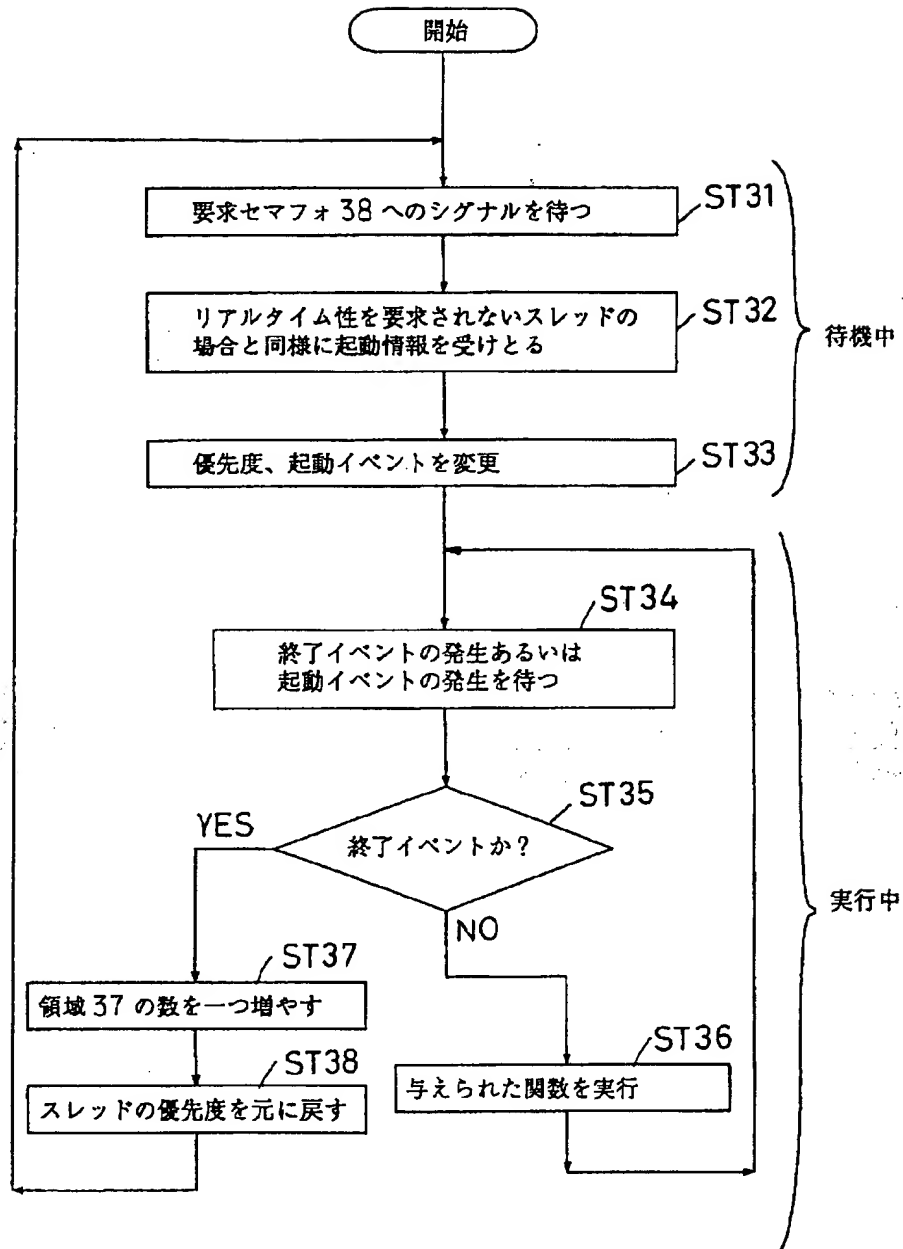
【図 9】



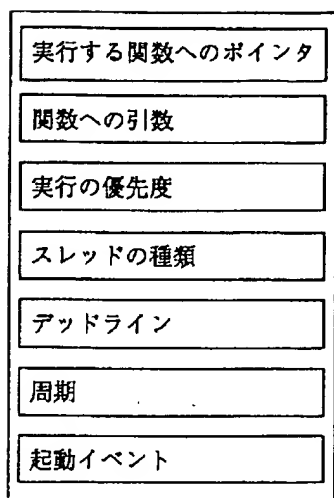
【図 23】



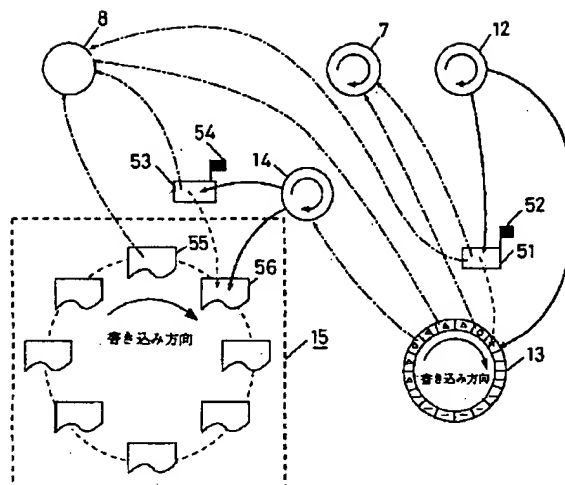
【図 11】



【図12】



【図14】

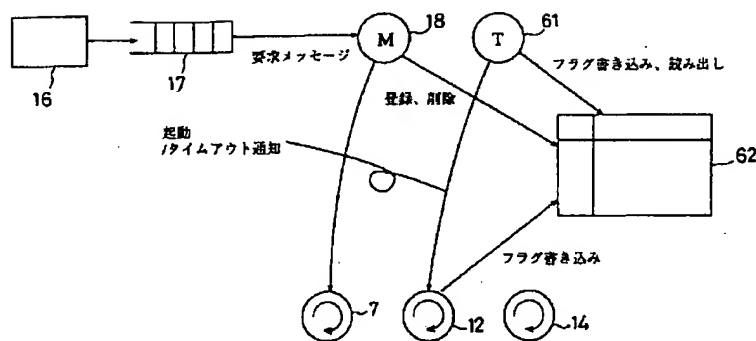


スレッドプール内のスレッドの起動情報

- × 書き込み中のインデックス
- 最新のデータを格納しているインデックス
- △ ディスク上に読み出し可能なバックアップのないインデックス
- ディスク上に読み出し可能なバックアップのあるインデックス

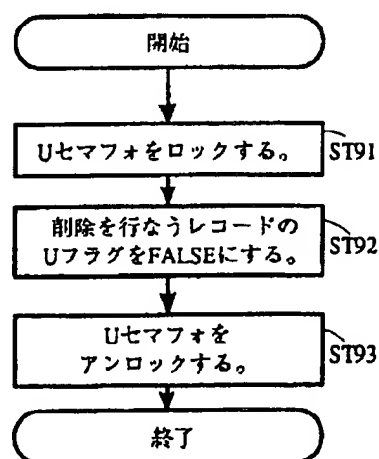
- リングバッファ、ディスク等への書き込みを表す。
- ← リングバッファ、ディスク等からの読み出しを表す。
- リングバッファのインデックスまたはファイルを表す。

【図15】

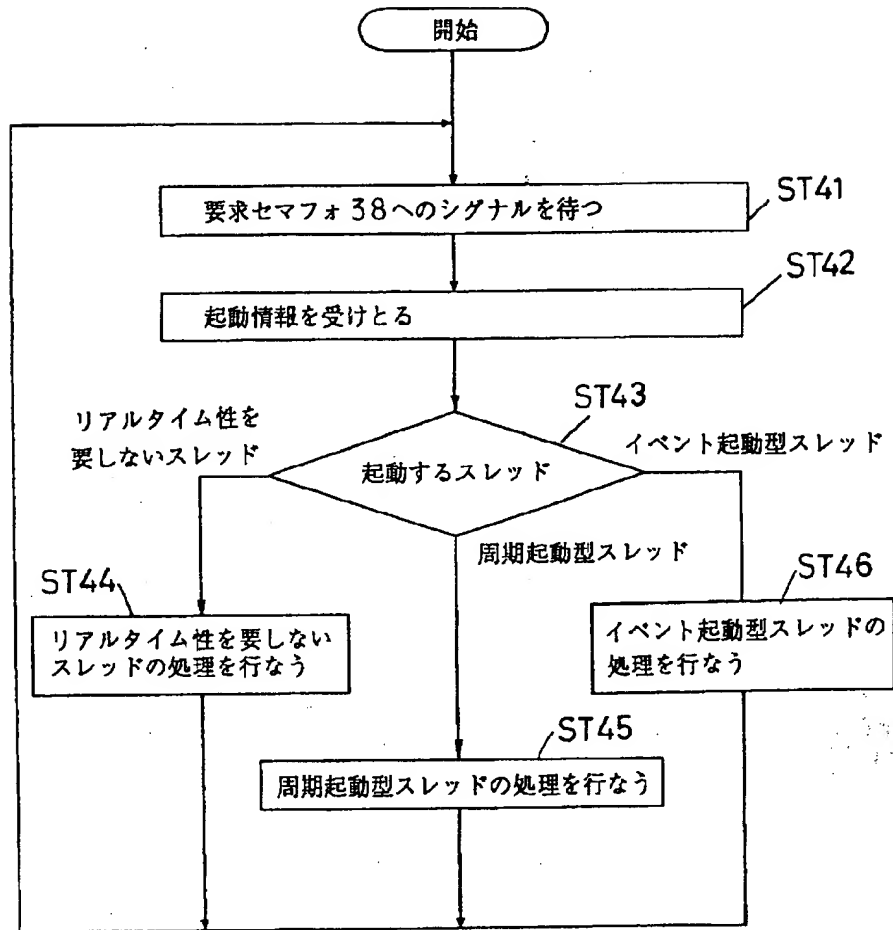


61: タイマースレッド
62: スレッド管理テーブル (資源)

【図19】



【図13】



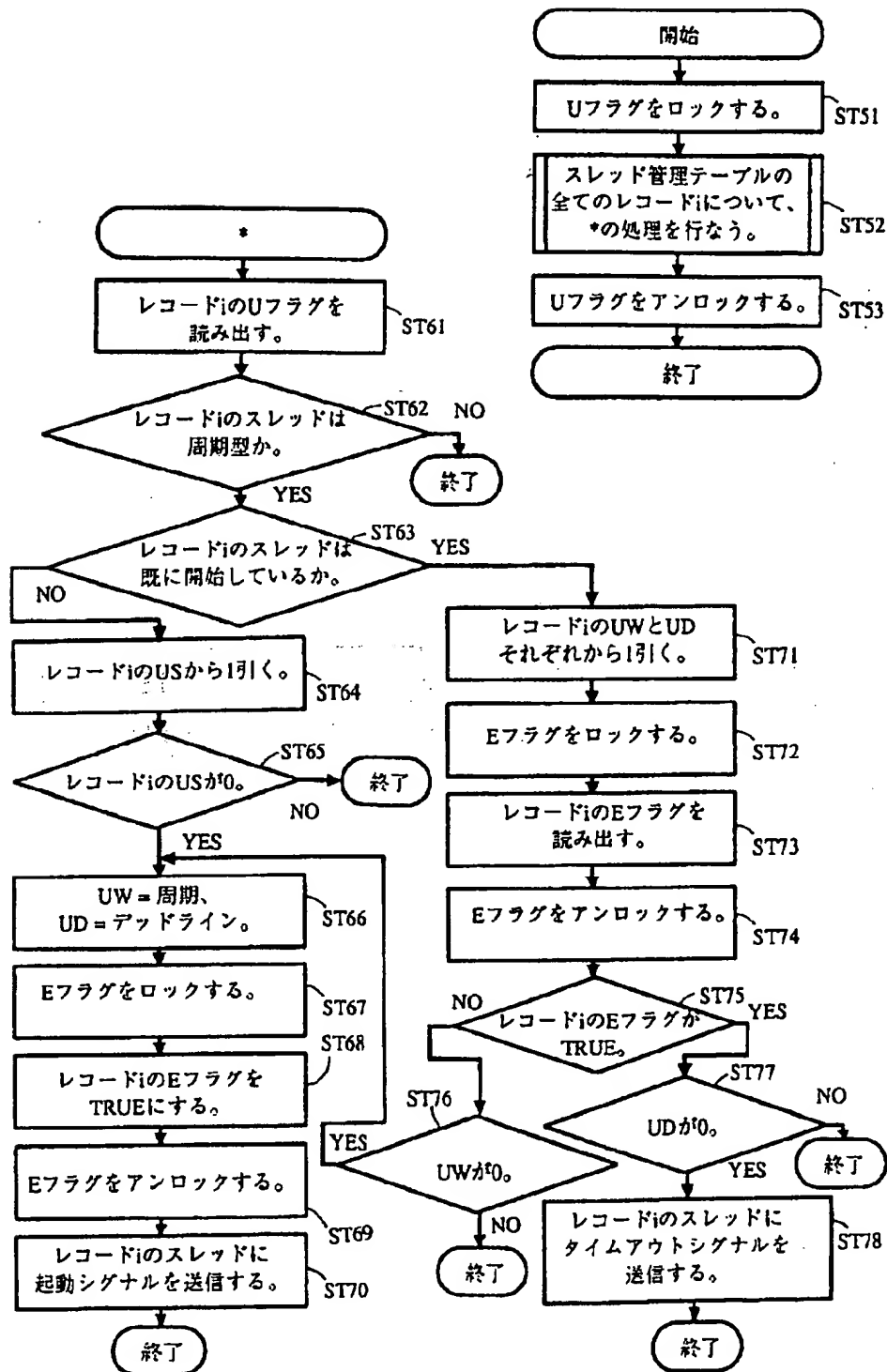
【図21】

インデックス	Uフラグ	Eフラグ	US	UW	UD	周期	デッドライン	Sフラグ
0	F	—						F
1	T	T						F
2	T	F						T
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
53	T	T						T
54	T	F						F
55	F	—						F
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮

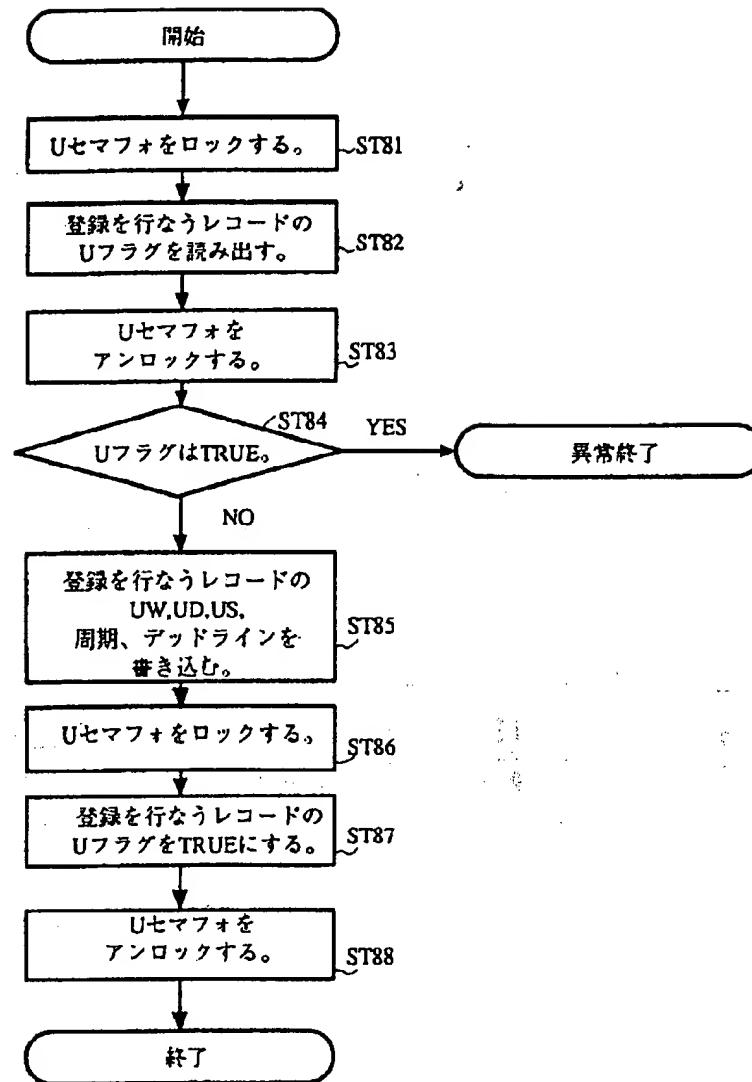
T : TRUE
F : FALSE
— : 任意

65: セマフォ

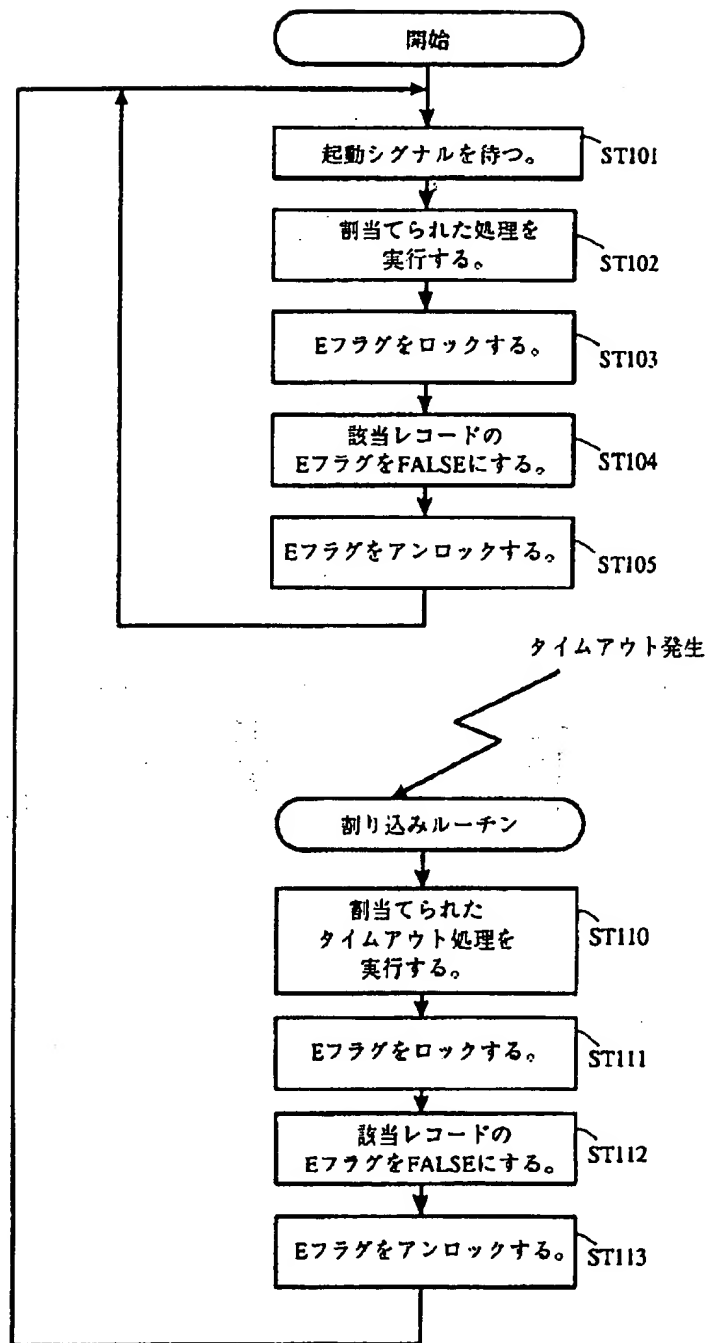
【図17】



【図18】



【図20】



【图 2 2】

